CLAIMS

What is claimed is:

1   1.   A method for storing unstructured XML data into a relational database, comprising:

2   assigning a document identifier to an XML document;

3   parsing the XML document to identify a node;

4   for the identified node in the XML document:

5   storing path information for the node;

6   storing hierarchical information for the node; and

7   storing node data for the node.

1   2.   The method of claim 1 in which the hierarchical information comprises a

2   hierarchical level within the XML document.

1   3.   The method of claim 1 in which the node data comprises a start position, end

2   position, node type, or node value.

1   4.   The method of claim 1 in which the document identifier is a unique identifier for

2   each different XML document.

1   5.   The method of claim 1 in which the path information comprises a full path for the

2   node.

1   6.   The method of claim 1 in which the path information comprises a path identifier.

1   7.   The method of claim 6 in which the path identifier corresponds to a key to a path

2   entry containing a full path for the node.

1   8.   The method of claim 7 in which the path entry resides in a first table structure and

2   the path information, hierarchical information, and node data reside in a second table

3   structure.

1   9.   The method of claim 7 in which the path entry comprises node name corresponding

2   to a name of a terminal node.

1   10.   The method of claim 1 further comprising:

2   maintaining one or more indexes.

1   11.   The method of claim 10 in which the one or more indexes comprise an index on a

2   path identifier, an index on the document identifier and a start position, or an index on the

3   document identifier, start position, and node level.

1   12.   The method of claim 10 in which the path identifier corresponds to a key to a path

2   entry containing a full path for the node, the path entry resides in a separate table, and the

3   one or more indexes comprise an index on path identifiers or a unique index on reverse

4   paths.

1   13.   A computer-implemented structure for storing XML data in a relational database, the

2   computer implemented structure comprising a first table structure, the first table structure

3   comprising:

4   a document identifier corresponding to an XML document;

5   path information for a node within the XML document;

6   hierarchical information for the node; and

7   node data for the node.

1   14.   The computer-implemented structure of claim 13 in which the hierarchical

2   information comprises a hierarchical level within the XML document.

1   15.   The computer-implemented structure of claim 13 in which the node data comprises

2   separate columns for a start position, end position, node type, or node value.

1   16.   The computer-implemented structure of claim 13 in which the document identifier is

2   a unique identifier for each different XML document.

1   17.   The computer-implemented structure of claim 13 in which the path information

2   comprises a full path for the node.

1   18.   The computer-implemented structure of claim 13 in which the path information

2   comprises a path identifier.

1   19.   The computer-implemented structure of claim 18 in which the path identifier

2   corresponds to a key to a path entry in a second table structure.

1   20.   The computer-implemented structure of claim 19 in which the path entry comprises a

2   full path for the node.

1   21.   The computer-implemented structure of claim 18 in which the path entry comprises a

2   node name corresponding to a name of a terminal node.

1   22.   A method to access a computer-implemented structure for storing XML data in a

2   relational database, the computer implemented structure comprising a first table structure,

3   the first table structure comprising a document identifier corresponding to an XML

4   document, path information for a node within the XML document, hierarchical information

5   for the node, and node data for the node, the method comprising:

6   generating a SQL query against the computer-implemented structure; and

7   producing a result set based upon executing the SQL query.

1   23.   The method of claim 22 in which the SQL query reconstructs the XML document.

1   24.   The method of claim 23 in which the SQL query provides the same result as the

2   following:

3   select i.nodename, p.startpos, p.endpos, p.nodetype, p.nodeval
4   from path_table p, path_index_table i
5   where p.docid = :1 and p.pid = i.pid

27

6   order by p.startpos

7   where path_table comprises a first column for the start position of the node (startpos), a

8   second column for the end position of the node (endpos), a node type column (nodetype), a

9   node value column (nodeval), a path identifier column (pid), and a document identifier

10   column (docid), and a path_index_table comprises a path identifier column (pid), a path

11   column (path), and a nodename column (nodename).

1   25.   The method of claim 22 in which the SQL query identifier a fragment within the

2   XML document.

1   26.   The method of claim 25 in which the SQL query provides the same result as the

2   following:

3

4   select i.nodename, p.startpos, p.endpos, p.nodetype, p.nodeval

5   from path_table p, path_index_table i,

6   (select docid, startpos, endpos from path_table

7   where rowid = :1) p2

8   where p.docid = p2.docid and p.startpos >= p2.startpos

9   and p.endpos <= p2.endpos and p.pid = i.pid

10   order by p.startpos

11

12   where path_table comprises a first column for the start position of the node (startpos), a

13   second column for the end position of the node (endpos), a node type column (nodetype), a

14   node value column (nodeval), a path identifier column (pid), and a document identifier

15   column (docid), and a path_index_table comprises a path identifier column (pid), a path

16   column (path), and a nodename column (nodename).

1   27.   The method of claim 22 in which the SQL query corresponds to an XPath

2   expression.

1   28.   The method of claim 27 in which the XPath expression is translated to the SQL

2   query by:

3   breaking the XPath expression into multiple components;

4   creating a new SQL query corresponding to each of the multiple components; and

5   joining the new SQL query corresponding a component to its previous component.

1   29.   The method of claim 28 in which the XPath expression is broken into multiple

2   components by considering each continuous segment of simple XPath, wherein each

3   occurrence of a predicate within the XPath causes creation of a new component.

1   30.   The method of claim 29 wherein a set of node names separated by "/" corresponds to

2   a single XPath component.

1   31.   The method of claim 28 in which the new SQL query comprises a join of a

2   path_index_table and a path_table.

1   32.   The method of claim 28 in which the new SQL query comprises one or more

2   conditions.

1   33.   The method of claim 32 in which the one or more conditions comprises a condition

2   for the path being chosen, a condition for the node type, or a condition for the node value.

1   34.   The method of claim 28 in which the act of joining the new SQL query

2   corresponding the component to its previous component uses a join condition comprising a

3   join on a document identifier or a join on a hierarchy relationship.

1   35.   A method for managing an unstructured document in a relational database system,

2   comprising:

3   storing the unstructured document in a storage structure in the relational database

4   system, the storage structure corresponding to a universal schema;

5   determining whether to create an index upon the storage structure, wherein one or

6   more indexes are maintained if desired; and

7   accessing the unstructured documents by accessing the storage structure.

1   36.   The method of claim 35 in which the unstructured document comprises an XML

2   document.

1   37.   The method of claim 36 in which the storage structure comprises:

2   a document identifier corresponding to an XML document;

3   path information for a node within the XML document;

4   hierarchical information for the node; and

5   node data for the node.

1   38.   The method of claim 37 in which the one or more indexes comprise an index on a

2   path identifier, an index on the document identifier and a start position, or an index on the

3   document identifier, start position, and node level.

1   39.   The method of claim 36 further comprising a second structure for storing path data,

2   the second structure comprising:

3   a path identifier;

4   a full path for the node; and

5   a node name corresponding to a name of a terminal node.

1   40.   The method of claim 39 in which the one or more indexes comprise an index on path

2   identifiers or a unique index on reverse paths.

1   41.   The method of claim 35 in which the unstructured documents are accessed by

2   accessing the storage structure using a SQL query.

1   42.   The method of claim 41 in which the SQL query reconstructs the XML document.

1   43.   The method of claim 41 in which the SQL query identifier a fragment within the

2   unstructured documents .

1   44.   The method of claim 41 in which an XPath expression is translated to the SQL query

2   by:

3   breaking the XPath expression into multiple components;

4   creating a new SQL query corresponding to each of the multiple components; and

5   joining the new SQL query corresponding a component to its previous component.

1   45.   A computer program product comprising a computer usable medium having

2   executable code to execute a process for storing unstructured XML data into a relational

3   database, the process comprising:

4   assigning a document identifier to an XML document;

5   parsing the XML document to identify a node;

6   for the identified node in the XML document:

7   storing path information for the node;

8   storing hierarchical information for the node; and

9   storing node data for the node.

1   46.   A system for storing unstructured XML data into a relational database, comprising:

2   means for assigning a document identifier to an XML document;

3   means for parsing the XML document to identify a node;

4   for the identified node in the XML document:

5   means for storing path information for the node;

6   means for storing hierarchical information for the node; and

7   means for storing node data for the node.

1   47.   A computer program product comprising a computer usable medium having

2   executable code to execute a process to access a computer-implemented structure for storing

3   XML data in a relational database, the computer implemented structure comprising a first

4   table structure, the first table structure comprising a document identifier corresponding to an

5   XML document, path information for a node within the XML document, hierarchical

6   information for the node, and node data for the node, the process comprising:

7   generating a SQL query against the computer-implemented structure; and

8   producing a result set based upon executing the SQL query.

1   48.   A system to access a computer-implemented structure for storing XML data in a

2   relational database, the computer implemented structure comprising a first table structure,

3   the first table structure comprising a document identifier corresponding to an XML

4   document, path information for a node within the XML document, hierarchical information

5   for the node, and node data for the node, the method comprising:

6   means for generating a SQL query against the computer-implemented structure; and

7   means for producing a result set based upon executing the SQL query.

1   49.   A computer program product comprising a computer usable medium having

2   executable code to execute a process for managing an unstructured document in a relational

3   database system, the process comprising:

4   storing the unstructured document in a storage structure in the relational database

5   system, the storage structure corresponding to a universal schema;

6   determining whether to create an index upon the storage structure, wherein one or

7   more indexes are maintained if desired; and

8   accessing the unstructured documents by accessing the storage structure.

1   50.   A system for managing an unstructured document in a relational database system,

2   comprising:

3   means for storing the unstructured document in a storage structure in the relational

4   database system, the storage structure corresponding to a universal schema;

5   means for determining whether to create an index upon the storage structure, wherein

6   one or more indexes are maintained if desired; and

7   means for accessing the unstructured documents by accessing the storage structure.

34